AUTONOMOUS NAVIGATION OF PLANETARY ROVERS: A FUZZY LOGIC APPROACH

Homayoun Seraji **Bruce Bon**
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109, USA

Abstract

This paper presents a new strategy for autonomous navigation of planetary rovers using the fuzzy logic framework and a novel on-board measure of terrain traversability. The navigation strategy is comprised of three simple, independent behaviors with different levels of resolution. The navigation rules for the first behavior, goal-seeking, utilize the global information about the goal position to generate the steering and speed commands that drive the rover to the designated destination. The navigation rules for the second behavior, terrain-traversing, use the regional information about the terrain quality to produce steering and speed commands that guide the rover toward the safest and the most traversable terrain. The inclusion of the regional terrain data (such as slope and roughness) in the rover navigation strategy is a major contribution of this paper. The navigation rules for the third behavior, collision-avoidance, employ the local information about the en-route obstacles to develop steering and speed commands that maneuver the rover around the encountered obstacles. The recommendations of these three behaviors are then integrated through appropriate weighting factors to generate the final control actions for the steering and speed commands that are executed by the rover. The weighting factors are produced by fuzzy rules that take into account the current status of the rover. The complete rover navigation strategy consists of a total of 37 fuzzy logic rules for the behaviors and their weighting factors. This navigation strategy requires no a priori information about the environment, and uses the on-board traversability analysis to enable the rover to select easy-to-traverse paths autonomously. The Rover Graphical Simulator developed at JPL for test and validation of the navigation rules, as well as for graphical visualization of the rover motion, is described. Three graphical simulation case studies are presented to demonstrate the capabilities of the proposed navigation strategy for planetary rovers. Finally, the navigation algorithm for the Sojourner rover is discussed and compared with the proposed strategy. Simulation studies clearly demonstrate the superior performance of this fuzzy navigation strategy relative to the Sojourner algorithm.

1 Introduction

In dealing with day-to-day processes, humans make subjective decisions based on qualitative information. Their perception of processes is based on qualitative, rather than quantitative, assessments obtained from low quality and approximate measurements. The human control strategy for a process typically consists of simple, intuitive rules based on prior experience that are brought to bear to affect the process. For instance, in the process of driving a car, the human driver turns the steering wheel to the right if the car veers too far to the left, and vice versa. The driver determines the degree of course correction intuitively and based on his driving experience, rather than relying on mathematical modeling and formulation of the process. His actions are based on how far from the lane the car has moved, and on how fast the car is moving. Similarly, the driver adjusts the speed of the car based on his subjective judgment of the road conditions. It is highly desirable to capture the expertise of the human driver and to utilize this knowledge to develop autonomous navigation strategies for planetary rovers. Fuzzy logic provides a means to accomplish this goal. Fuzzy logic provides a formal methodology for representing and implementing the human expert's heuristic knowledge and operational experience. Using the fuzzy logic framework, the attributes of the human reasoning and decision making can be formulated by simple *IF (antecedent) – THEN (consequent)* rules and easily-understandable *linguistic* representations. The linguistic values in the rule antecedents convey the imprecision associated with on-board sensor measurements; while those in the rule consequents represent the vagueness inherent in the reasoning processes. The operational strategies of the human expert driver can be transferred via fuzzy logic tools to the rover navigation strategy in the form of a set of simple conditional statements composed of linguistic variables. These linguistic variables are defined by fuzzy sets in accordance with user-defined membership functions. The main advantages of a fuzzy navigation strategy lie in the ability to extract heuristic rules from human experience and to obviate the need for an analytical model of the process. The natural appeal of fuzzy logic to rover navigation has motivated considerable research in this area in recent years [see, e.g., 1-30]. Most of this research, however, has been focused on indoor mobile robots that operate in highly structured, or otherwise, *man-made* environments.

NASA has planned an ambitious set of missions to launch a series of spacecraft to land on the planet Mars. These unmanned missions will carry mobile robots (rovers) to explore the Martian surface and to carry out science missions by placing instruments on or sampling from soil and rocks for analysis. The first in this series, the Mars Pathfinder mission that landed on Mars in July 1997, deployed the Sojourner rover on the Martian terrain for positioning science instruments against designated rocks, as shown in Figure 1. After the success of the Sojourner rover, there has been a strong motivation to develop future planetary rovers with enhanced capabilities that can explore remote planets autonomously and intelligently with minimal human intervention. The autonomy and on-board intelligence are particularly important for Mars missions where significant communication time-delay (up to 40 minutes round-trip) makes the teleoperation of rovers from Earth an impossible task. Therefore, on-board rover autonomy is a key enabling technology for future space exploration missions.

This paper develops a new strategy for autonomous navigation of *planetary* rovers using the fuzzy logic framework. Unlike conventional indoor mobile robots, these rovers are required to traverse harsh, *natural* terrains on remote planetary surfaces. The rover navigation strategy developed here is comprised of three simple behaviors: goal-seeking, terrain-traversing, and collision-avoidance. The fuzzy rules for the goal-seeking behavior use the *global* information about the goal position to make recommendations for the rover speed and steering. The fuzzy rules for the terrain-traversing behavior incorporate the *regional* information about the terrain to produce recommendations for the rover speed and steering. The fuzzy rules for the collision-avoidance behavior utilize the *local* information obtained from en-route obstacles to generate the rover speed and steering recommendations. Finally, the recommendations from the three behaviors are integrated with appropriate weighting factors to yield an autonomous navigation strategy for the planetary rover that requires no *a priori* information about the environment.

The paper is organized as follows. The rover navigation behaviors based on goal, terrain, and obstacle information are presented in Sections 2-4. The integration of these behaviors into a unified rover navigation strategy is discussed in Section 5. Sections 6 and 7 describe the Rover Graphical Simulator and present three computer simulation studies. The navigation algorithm of the Sojourner rover is discussed in Section 8 and compared with the proposed strategy. Finally, Section 9 reviews the paper and draws conclusions from this work.

2 Navigation Based on Global Goal Information

The problem considered in this paper is to safely navigate a rover on a planetary surface from a known initial position to a user-specified goal position. Figure 2 depicts the setting for this problem, where the position of the rover center is denoted by the coordinates (x, y) with respect to a fixed reference frame defined on the terrain[1], and the orientation of the rover body relative to the reference x-axis is represented by the angle θ, which is defined to be positive in the clockwise direction. The control variables of the rover are the translational speed v and the rotational speed (or turn rate) ω, where $v = \sqrt{(\frac{dx}{dt})^2 + (\frac{dy}{dt})^2}$ and $\omega = \frac{d\theta}{dt}$. The rover speed v is represented by the four linguistic fuzzy sets {STOP, SLOW, MODERATE, FAST }, with the membership functions shown in Figure 3a. Similarly, the rover turn rate ω is represented by the five linguistic fuzzy sets { FAST-LEFT, SLOW-LEFT, ON-COURSE, SLOW-RIGHT, FAST-RIGHT }, with the membership functions shown in Figure 3b.

In this section, we present fuzzy rules for navigation of the rover from its initial position to the desired goal position. Two sets of rules are developed for the rover speed v and the rover turn rate ω. The basic idea behind the goal-seeking navigation rules is that the rover tries to:

[1] Since the rover will always move on the three-dimensional terrain, its z-coordinate is dictated by the terrain geometry.

- Approach the goal with a speed proportional to the distance between the current position (x_c, y_c) and the goal position (x_g, y_g), defined as the "position error" d and computed on-line as (see Figure 2):

$$d = \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} \tag{1}$$

  The position error d is represented by the four linguistic fuzzy sets { VERY NEAR, NEAR, FAR, VERY FAR }, with the membership functions depicted in Figure 3c.

- Rotate toward the goal position by nullifying the "heading error" ϕ, which is the relative angle by which the rover needs to turn to face the goal directly (see Figure 2). The heading error is computed on-line as:

$$\phi = \theta_c - Atan2[(y_g - y_c), (x_g - x_c)] \tag{2}$$

  where θ_c is the current orientation of the rover. The heading error ϕ is represented by the five linguistic fuzzy sets { GOAL-FAR LEFT, GOAL-LEFT, GOAL-HEAD ON, GOAL-RIGHT, GOAL-FAR RIGHT }, with the membership functions depicted in Figure 3d.

Once the rover is sufficiently close to the goal, that is, $d \leq \epsilon$ where ϵ is a small user-specified threshold, the above definition of the heading error ϕ is changed to:

$$\phi = \theta_c - \theta_g \tag{3}$$

where θ_g is the desired goal orientation. With this change, the fuzzy rules cause an in-place rotation of the rover that will align the rover with the user-specified goal orientation at the destination[2].

We shall now present the fuzzy navigation rules for goal seeking in the following subsections.

2.1 Steering Rules

The rover turn rate ω depends on the heading error ϕ, where the angle ϕ is defined to be positive in the clockwise direction. The fuzzy rules for the rover turn rate are as follows:

- IF ϕ is GOAL-FAR LEFT, THEN ω is FAST-LEFT.
- IF ϕ is GOAL-LEFT, THEN ω is SLOW-LEFT.

[2]Note that, depending on the steering mechanism of the rover, the in-place rotation may or may not be achievable. For instance, this rotation is possible for skid-steering and Sojourner-type rovers but not for car-like rovers.

- IF ϕ is GOAL-HEAD ON, THEN ω is ON-COURSE.
- IF ϕ is GOAL-RIGHT, THEN ω is SLOW-RIGHT.
- IF ϕ is GOAL-FAR RIGHT, THEN ω is FAST-RIGHT.

It is seen that the rover turn rate ω is only a function of the heading error ϕ, and is independent of the rover speed v.

2.2 Speed Rules

The rover speed v is dependent on the position error d. The fuzzy rules for the rover speed are as follows:

- IF d is VERY NEAR, THEN v is STOP.
- IF d is NEAR, THEN v is SLOW.
- IF d is FAR, THEN v is MODERATE.
- IF d is VERY FAR, THEN v is FAST.

In addition, to decrease the rover turning radius when it is not pointed at the goal position, the following rule is added to the above speed rules:

- IF d is NOT VERY NEAR AND ϕ is NOT GOAL-HEAD ON, THEN v is SLOW.

The effect of this additional rule is to slow down the rover motion when it is not close to and not aligned with the goal.

3 Navigation Based on Regional Terrain Information

In a recent paper [31], the concept of Traversability Index is introduced as a simple measure for quantifying the suitability of a planetary surface for traverse by the rover under consideration[3]. This index is represented by the set of four linguistic fuzzy sets { POOR, LOW, MEDIUM, HIGH }, with the membership functions shown in Figure 4a. The Traversability Index is defined in terms of the terrain slope α and the terrain roughness β by a set of simple fuzzy relations summarized in Table 1 [31]. This index is computed on-board the rover using stereo cameras and associated software [32], and enables the rover to select easy-to-traverse terrains autonomously.

[3]The Traversability Index depends on the wheel design and traction mechanism of the rover which determine its hill climbing and rock climbing capabilities.

In this section, the Traversability Index is used to develop simple rules for determination of the rover steering and speed on a planetary surface. In other words, the Traversability Index is used to guide the rover toward the safest and the most traversable terrain. This index provides a simple means for incorporating the *regional* information perceived from the terrain quality data (out to about 10 meters) into the rover navigation strategy. The basic idea behind the terrain-traversing behavior is that the rover tries to:

- Rotate to face the terrain region with the highest value of Traversability Index, i.e., the safest and the most traversable terrain is selected.
- Adjust its speed based on the quality of the terrain to be traversed to avoid damaging the rover and ensure safety of rover motion.

We shall now discuss the fuzzy rules for determination of the rover turn rate and the rover speed based on the Traversability Index τ.

3.1 Steering Rules

It is assumed that the rover can only move in the forward direction (i.e., reverse motion is not allowed) and can turn in-place. The terrain in front of the rover is partitioned into five regions as shown in Figure 4b, namely: front, front-right, front-left, right, and left of the rover at a distance up to R from the rover, where R defines the radius of the sensing envelope and is typically 10 meters [32]. As shown in Figure 4b, "front" refers to the region directly ahead of the rover given its present heading, "front-right" and "front-left" regions are sectors between 0° and ±45° relative to the rover heading, and "right" and "left" regions are sectors between ±45° and ±90° relative to the heading. The Traversability Indices for the above five regions are computed from the measurements of the terrain slope and roughness that are obtained by the vision system on-board the rover [31, 32]. Therefore, at any instant, five crisp Traversability Indices are computed for the five possible traversable regions described above, namely: τ_l, τ_{fl}, τ_f, τ_{fr}, and τ_r. The on-board software then compares these five crisp quantities and selects the one with the highest value τ^*, that is, the most traversable region is chosen[4]. Let $\tau^* = Max\{\tau_l, \tau_{fl}, \tau_f, \tau_{fr}, \tau_r\}$. Then, the four turn rate rules are as follows:

- IF $\tau^* = \tau_l$, THEN ω is FAST-LEFT.
- IF $\tau^* = \tau_{fl}$, THEN ω is SLOW-LEFT.
- IF $\tau^* = \tau_{fr}$, THEN ω is SLOW-RIGHT.

[4]When the situation has a non-unique solution, i.e., there is more than one region with the highest τ, then the one which is closest to the front region is chosen so that unnecessary rotations are avoided. In the special cases when $\tau_r = \tau_l = \tau^*$ or $\tau_{fr} = \tau_{fl} = \tau^*$, in order to avoid confusion, additional rules should be added to steer the rover to the left as the preferred steering direction.

- IF $\tau^* = \tau_r$, THEN ω is FAST-RIGHT.

Note that since the antecedents of these rules are crisp, only one of the rules will fire and the consequent of this rule will be exclusively one of the output fuzzy sets. Observe that when $\tau^* = \tau_f$, i.e., the most traversable region is in front of the rover, the rover is already heading in the correct direction and no course change is necessary. Therefore, no new steering rule is needed in this case.

3.2 Speed Rules

Once the direction of traverse is chosen based on the relative values of τ, the rover speed v can be determined based on the value τ^* of the Traversability Index τ in the chosen region. This determination is formulated as a set of four simple fuzzy rules for speed of traverse as follows:

- IF τ^* is POOR, THEN v is STOP.
- IF τ^* is LOW, THEN v is SLOW.
- IF τ^* is MEDIUM, THEN v is MODERATE.
- IF τ^* is HIGH, THEN v is FAST.

Finally, for safety reasons, it is desirable to stop the rover motion momentarily when the rover needs to change its heading. To this end, the following rule is added for rover speed:

- IF $\tau^* \neq \tau_f$, THEN v is STOP.

In practice, other speed rules will fire as well and hence the rover will not make a complete stop. None-the-less, the effect of this rule is to slow down the rover for turning operations.

4 Navigation Based on Local Obstacle Information

In this section, fuzzy logic rules are described which govern the rover behavior based on the *local* information about the en-route obstacles, such as large rocks. This information is obtained on-line and in real-time by the proximity sensors mounted on the rover. Different types of proximity sensors can be used for this purpose, ranging from low-resolution infra-red sensors to high-resolution laser detectors [see, e.g., 33]. The range of operation of these local sensors is typically 10-30 cm, which is about two orders-of-magnitude smaller than that of regional sensors used in Section 3. Note that precise measurements of the obstacle distances are *not* needed, because of the multi-valued nature of the linguistic fuzzy sets used to describe them.

In the present analysis, it is assumed that there are three proximity sensors mounted on the rover facing the three different directions of front, right, and left. These sensors report the distances between the rover and the closest front obstacle d_f, the closest right obstacle d_r, and the closest left obstacle d_l within their ranges of operation. Furthermore, obstacles are sensed up to the sensing radius r, which is typically 10-30 cm. The three obstacle distances $\{ d_f, d_r, d_l \}$ are continously measured and updated during rover motion. The steering and speed rules use this local information to maneuver the rover around the obstacles and avoid potential collisions. Each obstacle distance d_f, d_r, or d_l is represented by the three linguistic fuzzy sets { VERY NEAR, NEAR, FAR }, with the membership functions shown in Figure 5. Note that we can have different definitions of these membership functions for front obstacle and side (left and right) obstacles so that front and side collisions will have different sensitivities.

The basic idea behind the obstacle avoidance rules is that the rover tries to:

- Turn to face a region with no nearby obstacles or with farther obstacles.
- Adjust its speed of motion depending on the distance to the closest front obstacle.

The obstacle avoidance navigation rules are discussed below.

4.1 Steering Rules

The goal of the steering rule set is to steer the rover clear of the obstacles. This goal is accomplished by sensing the three obstacle distances d_f, d_r, and d_l and reacting according to the following five fuzzy logic rules:

- IF d_f is NOT FAR AND d_l is FAR, THEN ω is SLOW-LEFT.
- IF d_f is NOT FAR AND d_l is NEAR AND d_r is FAR, THEN ω is SLOW-RIGHT.
- IF d_f is NOT FAR AND d_l is NEAR AND d_r is NEAR, THEN ω is SLOW-LEFT.
- IF d_f is NOT FAR AND d_l is NOT FAR AND d_r is VERY NEAR, THEN ω is SLOW-LEFT.
- IF d_f is NOT FAR AND d_l is VERY NEAR AND d_r is NOT VERY NEAR, THEN ω is SLOW-RIGHT.

The above rule set is summarized in Table 2 when d_f is NOT FAR. The following points are noted about the above steering rules. First, when d_f is FAR, i.e., the front of the rover is clear of obstacles, the rover will not collide with any obstacles and no corrective actions need to be taken. Therefore, the collision avoidance steering rules are activated only when the situation is otherwise. Second, the "preferred" direction of turn is taken to be LEFT, i.e., when the rover needs to turn to avoid an impending collision, it tends to turn left. The choice of LEFT instead of RIGHT is arbitrary, but selection of a preferred turn direction is essential to avoid the possibility that simultaneous left and right obstacles can result in a no-turn recommendation even though there may be an obstacle straight ahead.

4.2 Speed Rules

The speed rules for collision avoidance are very simple. Basically, the rover is required to slow down as it approaches the closest front obstacle. There are two fuzzy rules as follows:

- IF d_f is VERY NEAR, THEN v is STOP.
- IF d_f is NEAR, THEN v is SLOW.

Again, note that when the front obstacle distance is FAR, collision avoidance is not activated and no corrective actions need to be taken.

5 Integration of Seek, Traverse, and Avoid Behaviors

In the preceding sections, we discuss three *individual* behaviors of goal-seeking, terrain-traversing, and collision-avoidance. Each behavior accomplishes a single objective within a restricted context. The three behaviors operate *independently* of one another and generate recommendations based on the sensed data (obtained from hardware or software sensors) and the desired objective. The behaviors also operate at three different ranges of information, with the goal-seeking behavior at *global*, the terrain-traversing behavior at *regional*, and the collision-avoidance behavior at *local* ranges. In this section, these three behaviors are integrated to form a unified autonomous navigation strategy for planetary rovers without *a priori* knowledge about the environment. The approach adopted here for behavior integration proceeds in two stages. In the first stage, the goal-seeking, terrain-traversing, and collision-avoidance behaviors make their individual, independent recommendations for rover navigation. In the second stage, these recommendations are integrated by using appropriate weighting factors to generate the combined, coordinated control actions for the rover navigation based on the rover status.

Consider the rover navigation strategy shown in the block diagram of Figure 6a. Each of the three behaviors generates a set of independent recommendations for the translational and rotational speeds v and ω based on its fuzzy rules discussed in Sections 2-4. These sets of recommendations are denoted by $\{v^s\}, \{\omega^s\}$, $\{v^t\}, \{\omega^t\}$, and $\{v^a\}, \{\omega^a\}$, where the superscripts s, t, and a refer to the seek, traverse, and avoid behaviors, respectively. These recommendations are then "weighted" by the crisp weighting factors s^w, t^w, and a^w assigned to the outputs of the goal-seeking, terrain-traversing, and collision-avoidance behaviors, respectively. In other words, the final control actions $\bar{v}$ and $\bar{\omega}$ result from defuzzification of the weighted aggregated outputs of the goal-seeking, terrain-traversing, and collision-avoidance rule sets. Mathematically, the final control actions are computed using the Center-of-Gravity defuzzification method [34] as:

$$\bar{v} = \frac{s^w \Sigma v_p^s . A_p^s + t^w \Sigma v_p^t . A_p^t + a^w \Sigma v_p^a . A_p^a}{s^w \Sigma A_p^s + t^w \Sigma A_p^t + a^w \Sigma A_p^a} \tag{4}$$

$$\bar{\omega} = \frac{s^w \Sigma \omega_p^s . B_p^s + t^w \Sigma \omega_p^t . B_p^t + a^w \Sigma \omega_p^a . B_p^a}{s^w \Sigma B_p^s + t^w \Sigma B_p^t + a^w \Sigma B_p^a} \tag{5}$$

In the above equations, v_p and A_p are the peak value and the truncated area under the membership function for the velocity fuzzy sets, while ω_p and B_p are the corresponding values for the turn rate fuzzy sets.

The weighting factors s^w, t^w, and a^w represent the strengths by which the goal-seeking, terrain-traversing, and collision-avoidance recommendations are taken into account to compute the final control actions $\bar{v}$ and $\bar{\omega}$. These factors are represented by the three linguistic fuzzy sets {LOW, NOMINAL, HIGH }, with the triangular membership functions with peak values of 0, 1 and, 10, respectively, as shown in Figure 6b. Three sets of weight rules for the three behaviors are now presented. The goal-seeking weight rules are as follows:

- IF d is VERY NEAR, THEN s^w is HIGH.
- IF d is NOT VERY NEAR, THEN s^w is NOMINAL.
- IF τ^* is POOR OR τ^* is LOW, THEN s^w is LOW.
- IF d_f is NOT FAR, THEN s^w is LOW.

The terrain-traversing weight rules are as follows:

- IF d is VERY NEAR, THEN t^w is LOW.
- IF d is NOT VERY NEAR AND $\tau_f \neq \tau^*$, THEN t^w is HIGH.
- IF d is NOT VERY NEAR AND τ^* is POOR OR τ^* is LOW, THEN t^w is HIGH.
- IF τ^* is NOT POOR AND τ^* is NOT LOW AND $\tau_f = \tau^*$, THEN t^w is NOMINAL.

Finally, the collision-avoidance weight rules are as follows:

- IF d is VERY NEAR, THEN a^w is LOW.
- IF d is NOT VERY NEAR AND d_f is FAR, THEN a^w is LOW.
- IF d is NOT VERY NEAR AND d_f is NOT FAR, THEN a^w is HIGH.

At any control cycle, the above sets of weight rules are used to calculate the three crisp weighting factors using the Center-of-Gravity defuzzification method [34]. The resulting crisp weighting factors are then used to compute the final control actions for the rover speed and turn rate that are executed by the rover. Note that the $\{v\}$ and $\{\omega\}$ recommendations from each behavior are *independent* of other behaviors, however, the weighting factors are *not* independent since they control behavior interactions. The complete rover navigation strategy consists of a total of 37 fuzzy logic rules for the behaviors and their weighting factors. Finally, to avoid sudden variations in the speed and steering commands, the crisp control actions for $\bar{v}$ and $\bar{\omega}$ are averaged as:

$$\bar{v}_n = \frac{\bar{v}_n + \bar{v}_{n-1}}{2} \tag{6}$$

$$\bar{\omega}_n = \frac{\bar{\omega}_n + \bar{\omega}_{n-1}}{2} \tag{7}$$

where the subscripts n and $n-1$ denote the present and the previous sampling instants. This averaging operation smoothes out sudden changes in the control actions and prevents jerky motions of the rover.

6 Rover Graphical Simulator

The Rover Graphical Simulator (RGS) provides an essential tool for visualization of the rover motion using the reasoning and decision making capabilities provided by the fuzzy logic navigation strategy developed in this paper. RGS is written as a Java applet for platform independence, and runs on a PC as well as on Sun and SGI Unix machines.

A snapshot of the principal RGS window is shown in Figure 7a. It consists of:

- A large central panel containing a simple, two-dimensional graphical simulation depicting a terrain with a rover moving among obstacles and regions of various traversability indices.
- A lower panel containing buttons for selecting pre-stored test cases and navigation algorithms, and for turning on or off interactive placement and orientation of the initial and the goal rover configurations (i.e., positions and orientations).
- An upper panel displaying duration of the current or the most recent execution, and a set of algorithm-dependent buttons.

In the graphical simulation panel, registration dots are displayed every 100 pixels – the coordinates used in the simulation are in units of screen pixels. For each of the twelve pre-stored test cases, RGS shows the initial and the goal rover configurations as outlines and, when execution is started, animates the rover moving from the initial configuration toward the goal configuration. Both positions and orientations of the initial and the goal rover configurations can be changed interactively using the mouse buttons. This feature enables the user to create any desired scenario in real-time and to test out various elements of the navigation strategy. The rover motion is a discrete numerical simulation with a constant delta-time between updates. At each simulation step, a dot is placed at the old rover position and the graphical rover is redrawn at the new, updated rover position. This has the effect of leaving a trail of dots behind the rover which depicts the path traversed by the rover, while the spacings between the dots indicate the rover speed. Obstacles and regions of poor or low traversability are depicted as color-coded filled circles, with buffer zones around them indicated by green enclosing circles.

RGS uses the navigation algorithm selected by the user to drive an animation of the rover moving from the initial configuration toward the goal configuration. The navigation algorithms currently implemented include: (1) simple goal-seeking with no collision avoidance, (2) goal-seeking with virtual force repulsion for avoidance of obstacles, analogous to [35], (3) fuzzy logic navigation with selectable rule sets for goal-seek, traverse-terrain, avoid-collision, and behavior weights described in Sections 2-5, and (4) the Sojourner navigation algorithm used on the Mars Pathfinder mission, adapted for RGS. For all of these algorithms, the rover is constrained to make no sideway motions; at each simulation step, it moves forward by a distance computed by the selected navigation algorithm, and then turns by an angle computed by the navigation algorithm.

Figure 7b is a screen snapshot showing the Execution Control Graphical User Interface (GUI) window on the left and the Rule Strength Bar Chart window on the right. The buttons in the Execution Control window allow the user to start execution of a test case, to pause or continue execution, to single-step execution from a pause, to abort execution of a test case, and to speed up or slow down the simulation relative to real-time. The DEBUG button turns on a flag which generates diagnostic data in the start-up window, and the Log Crisp Inputs/Outputs button prints data during a simulation to the start-up window. The Stand-off Distance field allows user setting of the stand-off distance, which is used to create a buffer zone around both avoidance obstacles and low-traversability regions. The Rule Strength Bar Chart is a dynamic graphical display of the levels at which the 37 fuzzy logic rules are fired at any instant in time. Execution of any test case brings up a real-time bar chart display of the firing strengths of all the rules, as shown in Figure 7b.

All of the functionalities for supporting rover navigation simulations are embedded within the RGS application. In addition to the graphical rover animation and the Execution Control, this includes a built-in fuzzy logic engine for implementation of fuzzy logic rules, and linguistic fuzzy sets with user-defined membership functions. For fuzzy logic navigation, the rule sets for the three behaviors (seek-goal, traverse-terrain, and avoid-collision) and the rule sets for the weighting factors are integrated in the fuzzy logic engine. The algorithm-dependent buttons in the top panel of the principal window (Figure 7a) allow the user to select particular rule sets for the behaviors and the weights, and to printout the current fuzzy rules.

7 Simulation Studies

In this section, graphical simulation results are presented to demonstrate the fuzzy-based rover navigation strategy developed in this paper. The simulations are performed using the Rover Graphical Simulator (RGS) described in Section 6. Three case studies are presented in this section. In all cases, the terrain has HIGH Traversability Index, unless stated otherwise.

7.1 Case Study One

In this case, there is a large rock and a crater with POOR Traversability Index between the initial and the goal positions of the rover, as depicted in Figure 8a. The autonomous navigation strategy is required to drive the rover to the goal position while avoiding the impassable crater and preventing collision with the rock. Two circular safety regions are also defined which are displaced from the crater and the rock by user-specified stand-off distances. The path traversed by the rover under the fuzzy navigation rules is shown by the dotted line in Figure 8a. It is seen that the test is successfully completed with the rover reaching the goal safely, while avoiding both the crater and the rock.

7.2 Case Study Two

In this case, there are two large rocks and a crater with POOR Traversability Index between the initial and the goal positions of the rover, as depicted in Figure 8b. The rover is required to drive to the goal position while avoiding both rocks and the crater. Figure 8b depicts the path traversed by the rover. It is seen that the test is successfully completed with the rover reaching the goal safely, and both rocks and the crater are avoided.

7.3 Case Study Three

In this case, there are two large rocks, a crater with POOR Traversability Index, and a region of high rock density with LOW Traversability Index between the initial and the goal positions of the rover, as depicted in Figure 8c. The rover is required to drive to the goal position while avoiding both rocks and both impassable regions. Each rock and impassable region is surrounded by a user-defined safety zone. The path traversed by the rover under the fuzzy navigation rules is shown by the dotted line in Figure 8c. It is seen that the test is successfully completed with the rover reaching the goal safely, while avoiding both rocks and the two impassable regions.

8 Comparison with Sojourner Navigation Algorithm

When the Pathfinder spacecraft landed on Mars in July 1997, the Sojourner rover emerged from the lander, taking pictures from the Martian terrain and positioning science instruments against designated rocks. The Sojourner rover is approximately 68 cm long by 48 cm wide, with a fully deployed height of 28 cm [36]. Including all instruments and telecommunications equipment, the rover has a mass of about 17 kilograms. Mobility is provided by a 6-wheel drive rocker-bogie mechanism, with the two front and the two rear wheels independently steerable, allowing both conventional steering during forward or reverse motion as well as skid-steering for turn-in-place maneuvers. Forward speed in nominal terrain is 0.4 m/min or about 7 mm/sec.

The operational procedure of the Sojourner rover is now described briefly. The path planning portion of the rover mission is carried out manually by the mission operator on Earth. The operator designates a path to the goal location by specifying 3-D way-points for the rover using stereo imagery from the stationary lander. Way-points are specified using a stereo-graphic display to view the scene and a space-ball to input 3-D coordinates. The rover path is then generated by connecting the way-points using straight-line segments. For autonomous navigation between the way-points, the Sojourner rover employs a simple method based on the behavior control approach [37, 38]. This approach uses very simple steering logic based on the instantaneous state of the rover hazard detection sensors, and does not use an internal map or memory of previously encountered hazards. Basically, the navigation logic is as follows [39, 40]:

```
IF there is no hazard,
THEN move forward and turn toward the goal,
ELSE IF there is a hazard on the left,
THEN turn in place to the right until no hazard is detected,
ELSE IF there is a hazard on the right,
THEN turn in place to the left until no hazard is detected.
```

Additionally, hazards in the center are avoided by turning right if that is clear, otherwise turning left. The Sojourner rover also has a limited ability to navigate between hazards and to back away from dead-ends. The simplicity of this navigation algorithm makes it practical to implement on the Intel 8085 rover flight processor embedded in Sojourner. This simplistic approach can work effectively provided that the terrain is sparsely populated with rocks.

The primary sensor data used for the Sojourner navigation is provided by a combination of cameras and lasers that project vertical stripes in 5 forward directions. After software analysis of the images, the resulting data consists of 20 elevation estimates in a fan-shaped pattern ahead of the rover, corresponding to 4 ranges for each of the 5 laser stripe directions. This sensor data ranges from 56 cm to 81 cm in front of the rover. Sensor scans are only performed when the rover is stationary following a turn-in-place maneuver, a short (6.5 cm) forward traverse, or a backup away from a hazard. A hazard is identified as either an elevation datum that is above a certain threshold, or a missing sensor datum that may indicate a drop-off ahead. If Sojourner detects such a hazard, it turns in place by the fixed amount of 45° away from the hazard and does another sensor scan. If it still detects a hazard in the same direction, it will turn another 45° and so on until it has a clear sensor scan. If it detects a hazard in the opposite direction, it will turn halfway back to the former heading and try to squeeze through. If this fails, with detection of a hazard straight ahead, it will backup and turn again, and if this still fails to find a clear path, it will give up and report failure to ground control. The Sojourner rover navigation capability is limited by the low-resolution, noisy data provided by its short-range sensors, and by the lack of longer range sensor data, as well as by the simplistic capabilities of its behavior control implementation.

The pre-flight performance of the Sojourner rover is analyzed and evaluated extensively in [39]. It is pointed out that the operator-based path planning method used is subject to two types of errors. Way-point coordinates are subject to systematic errors caused by miscalibration of the cameras and pan/tilt axes, and also to random errors caused by using estimation of the stereo-disparity of the way-points. Systematic errors, in the form of translational and rotational errors in the location of the camera in the lander coordinate frame, compound the way-point designation uncertainty. The performance analysis carried out in [39] predicts that the combined designation error will exceed the dimension of the vehicle for distances beyond 20 meters away from the lander, making it difficult for the rover to find the target rocks at such ranges based on the designated-way-point path planning method. On the navigation side, the simplistic navigation algorithm described above is severely restricted for collision avoidance in areas commonly found on Mars that are densely populated by rocks.

The terrain navigation behavior introduced in [31] and described in section 3 allows the rover navigation system to look ahead and to select the most traversable path. The only similar capability implemented on the Sojourner rover is the ability to detect the slope in proximity of the rover from differences in sensor elevation data. Sojourner's response to encountering a region of unacceptable slope is to turn away from the region. The proposed strategy, however, will see a region of low traversability and decide either to turn away from it before encountering it, or to continue through the region at a lower speed.

In the Sojourner rover, all navigation behaviors are implemented directly in 1300 lines of C code which obscures the underlying logic, and the conditions for transitioning between the different behaviors are complex and distributed within the code. In contrast, the proposed fuzzy logic-based algorithm is embodied in a total of 37 if-then rules that control all three behaviors and the weights that are used to blend the behaviors[5]. The fuzzy rules are far easier to comprehend in order to modify or tune the algorithm, and it is easier to add new rules to provide more sophisticated behaviors.

Another advantage derived from use of fuzzy logic is a smooth and effective blending of the behaviors. In the Sojourner rover, if one of the behaviors is active, the other behaviors are completely turned off. Thus, when a transition between the behaviors occurs, it is characterized by an abrupt and discontinuous change in the movement of the rover. The Sojourner speed is very slow (7 mm/sec), but future rovers with more ambitious scientific

[5]Note that there exists an underlying fuzzy logic engine, comprised of many lines of code, which interprets the fuzzy rules, but this complexity is not visible to the algorithm designer. The point of comparison is *not* that the fuzzy algorithm requires fewer lines of code, but rather that the much higher level of abstraction of fuzzy logic facilitates creation, understanding, and modification of the navigation algorithm.

missions will be faster and such abrupt changes in movement may become a problem. In the fuzzy navigation strategy, the behaviors are blended smoothly using the weights described in Section 5. A rover controlled by this system moves smoothly toward the goal while avoiding obstacles and seeking paths of high traversability, as illustrated in Section 7.

In many cases, the Sojourner navigation algorithm executes a successful local path to a specified goal (i.e., way-point) location, while avoiding collision with en-route obstacles. In other fairly simple cases, the Sojourner algorithm fails; Figure 9 illustrates such a situation. In Figure 9a, the proposed fuzzy navigation strategy finds a smooth and efficient path from the initial to the goal locations, while achieving collision avoidance. Figure 9b illustrates two samples of rover positions achieved by the Sojourner algorithm, in which the rover alternately turns right, away from one rock, and then left to avoid the other, never achieving the goal position. Initially the rover begins to curve toward the goal location represented by the isolated outline rover, but then it encounters the oval-shaped rock. After failing to squeeze between the rocks, it turns to the right, but then it sees the other rock. It reacts by turning back to the left, but then it sees the first rock again, and the cycle repeats itself until the rover gives up and declares failure.

Another example in which the Sojourner navigation algorithm fails occurs when the rover encounters a poor terrain. In these situations, the Sojourner's behavior is very rudimentary. It turns in 45° increments away from the current heading in the direction of the goal position (right or left) until its sensors no longer detect poor terrain; it then drives straight for 40 cm before reverting to its goal-seeking behavior. Because the Sojourner algorithm makes no distinction based on where the region of poor terrain is detected, this can lead to oscillatory behavior, as illustrated in Figure 10 which shows the *same* test case used for Case Study One (Section 7.1, Figure 8a). When it first detects the crater, the rover turns away from it to the left, drives straight for a short distance, then starts curving right (toward the goal), which makes it encounter the crater again. This time the rover turns right (toward the goal) and keeps turning right until it has a clear sensor sweep. Then it drives straight for a short distance and starts curving left (toward the goal), which again makes it encounter the crater. This cycle will continue indefinitely[6].

The conclusions drawn from this comparative study are summarized below:

- The collision avoidance feature on the Sojourner rover fails in many cases where there are clearly collision-free paths for the rover to traverse. This a consequence of the simplistic nature of its behavior-based navigation algorithm. The fuzzy navigation strategy, on the other hand, has proven to be robust and reliable for collision avoidance.

- The addition of sensors capable of measuring the physical properties of the terrain, such as slope and roughness, out to about 10 meters, along with fuzzy navigation rules to

[6]In fairness to Sojourner, it should be noted that the oscillatory behavior illustrated in Figures 9 and 10 is significantly different on the real rover due to noisy sensor data, wheel slippage, irregular natural terrain, and many other factors that give the rover a degree of pseudo-random behavior which might ultimately allow it to get past obstacles or regions of poor terrain.

take advantage of this sensor data, provide a significant new capability for autonomous navigation on natural and unpredictable planetary surfaces. This capability simply does not exist in the Sojourner navigation algorithm, where the range of terrain sensing is limited to only 81 centimeters. As a consequence, the Sojourner algorithm can lead the rover to an impassable terrain or a failure, even when a safe path exists and is traversed by the fuzzy algorithm.

We conclude that the fuzzy navigation strategy developed in this paper yields superior performance in comparison with the Sojourner navigation algorithm.

9 Conclusions

Exploration of planetary surfaces by mobile robots offers several technical challenges. Planetary rovers must be able to operate autonomously and intelligently with minimal interaction with Earth-based operators. To accomplish this goal, the rovers must have the on-board intelligence needed for long-range traverse in highly-unstructured, poorly-modeled terrains with a high level of robustness and reliability. Their on-board intelligence must be capable of real-time navigation and motion planning based on poor and noisy sensor data. Fuzzy logic provides a natural framework for expressing the human reasoning and decision making processes for driving the rover on a planetary surface. The human driving strategy can then be transferred easily to the on-board navigation system for planetary rovers.

Rover navigation strategies based on fuzzy logic have major advantages over analytical methods. First, the fuzzy rules that govern the rover motion are easily understandable, intuitive, and emulate the human driver experience. Second, the fuzzy strategy can be extended very easily to incorporate new constraints and new criteria – whereas this requires complete reformulation for analytical methods. And third, fuzzy navigation allows integration of multiple behaviors into a unified strategy, together with smooth interpolation between the behaviors to avoid abrupt and discontinuous transitions.

Although the proposed fuzzy navigation strategy shares a common architecture with some of the references cited here, the fuzzy logic rules for the goal-seeking and collision-avoidance behaviors and the manner in which the behaviors are blended are different. In addition, the terrain-traversing behavior is a new feature that is introduced for the first time in this paper. It is noted that the graphical simulation environment described in Section 6 assumes that the rover position is known exactly and the obstacle and terrain sensor data are noise-free. In practice, rover position measurement has some error and sensor data is contaminated with noise. Therefore, it is necessary to conduct laboratory tests on a real rover to validate any navigation strategy. However, given the multi-valued nature of fuzzy logic, this approach is expected to have a reasonable level of robustness in practice.

The autonomous navigation strategy developed in this paper operates at three levels of resolution which are typically one to two orders-of-magnitude apart. First, the *global* information about the goal position which is typically up to 100 meters away. Second, the *regional* information about the terrain which is typically up to 10 meters away. And third, the *local* information about the obstacles which is typically up to 30 centimeters away. A novel contribution of this paper is the utilization of the terrain quality data represented by the Traversability Index in the rover navigation strategy. The Traversability Index embodies the information about the terrain slope and roughness obtained by sensors mounted on the rover. The on-board traversability analysis enables the rover to select easy-to-traverse paths autonomously. The terrain-traversing behavior introduced in this paper is analogous to the human actions in driving a car, where the car speed and steering are adjusted continuously based on the conditions of the road. The autonomous navigation strategy developed in this paper is currently being implemented on a commercial outdoor mobile robot for test and evaluation.

10 Acknowledgments

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. Technical discussions with Jack Morrison of TELOS and Edward Tunstel of JPL on the Sojourner navigation algorithm and on fuzzy logic are gratefully acknowledged.

11 References

1. A. Saffiotti: "The uses of fuzzy logic in autonomous robot navigation", Journal of Soft Computing, vol. 1, no. 4, pp. 180-197, 1997.

2. M. Sugeno and M. Nishida: "Fuzzy control of a model car", Fuzzy Sets and Systems, no. 16, pp. 103-113, 1985.

3. M. Sugeno, T. Murofushi, T. Mori, T. Tatematsu, and J. Tanaka: "Fuzzy algorithmic control of a model car by oral instructions", Fuzzy Sets and Systems, no. 32, pp. 207-219, 1989.

4. M. Sugeno and K. Murakami: "Fuzzy parking control of a model car", Proc. IEEE Conf. on Decision and Control (CDC), pp. 902-903, Las Vegas, 1984.

5. C. von Altrock, B. Krause, and H. J. Zimmermann: "Advanced fuzzy logic control of a model car in extreme situations", Fuzzy Sets and Systems, no. 48, pp. 41-52, 1992.

6. E. H. Ruspini: "Fuzzy logic in the Flakey robot", Proc. Intern. Conf. on Fuzzy Logic and Neural Networks, pp. 767-770, Iizuka (Japan), 1990.

7. J. L. Martinez, A. Ollero, and A. Garcia-Cerezo: "Fuzzy strategies for path tracking of autonomous vehicles", Proc. First European Congress on Fuzzy and Intelligent Technologies, pp. 24-30, Aachen (Germany), 1993.

8. C. Isik: "Identification and fuzzy rule-based control of a mobile robot motion", Proc. IEEE Intern. Symposium on Intelligent Control, pp. 94-99, 1987.

9. M. Maeda, Y. Maeda, and S. Murakami: "Fuzzy drive control of an autonomous mobile robot", Fuzzy Sets and Systems, no. 39, pp. 195-204, 1991.

10. L. X. Wang: "A mathematical formulation of hierarchical systems using fuzzy logic systems", Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 183-188, Orlando, 1994.

11. H. Surmann, J. Huser, and L. Peters: "A fuzzy system for indoor mobile robot navigation", Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 83-88, Yokohama (Japan), 1995.

12. K. Demirli and I.B. Turksen: "Mobile robot navigation with generalized modus ponens type fuzzy reasoning", Proc. IEEE Conf. on Systems, Man, and Cybernetics, pp. 3724-3729, 1995.

13. F. G. Pin, H. Watanabe, J. R. Symon, and R. S. Pattay: "Using custom-designed VLSI fuzzy inferencing chips for the autonomous navigation of a mobile robot", Proc. IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems, pp. 790-795, Raleigh, 1992.

14. F. G. Pin, H. Watanabe, J. R. Symon, and R. S. Pattay: "Autonomous navigation of a mobile robot using custom-designed qualitative reasoning VLSI chips and boards", Proc. IEEE Intern. Conf. on Robotics and Automation, pp. 123-128, Nice (France), 1992.

15. F. G. Pin and Y. Watanabe: "Driving a car using reflexive fuzzy behaviors". Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 1425-1430, San Francisco, 1993.

16. J. Z. Sasiadek and Q. Wang: "3-D guidance and navigation of mobile and flying robot using fuzzy logic", Proc. AIAA Guidance, Navigation, and Control Conf., pp. 1-9, Boston, 1998.

17. R. A. Hogle and P. P. Bonissone: "A fuzzy algorithm for path selection in autonomous vehicle navigation", Proc. IEEE Conf. on Decision and Control (CDC), pp. 898-900, Las Vegas, 1994.

18. R. N. Lea, Y. Jani, M. G. Murphy, and M. Togai: "Design and performance of a fuzzy logic based vehicle controller for autonomous collision avoidance", Proc. Fuzzy Neural Systems and Vehicle Applications Conf., Tokyo (Japan), 1991.

19. A. Saffiotti, E. H. Ruspini, and K. Konolige: "Blending reactivity and goal-directedness in a fuzzy controller", Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 134-139, San Francisco, 1993.

20. J. W. Baxter and J. R. Bumby: "Fuzzy logic guidance and obstacle avoidance algorithms for autonomous vehicle control", Proc. IFAC Workshop on Intelligent Autonomous Vehicles, pp. 257-262, Southampton (England), 1993.

21. W. Li: "Fuzzy logic-based perception-action behavior control of a mobile robot in uncertain environments", Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 1626-1631, Orlando, 1994.

22. W. Blochl: "Fuzzy control in real-time for vision guided autonomous mobile robots", Proc. Austrian Conf. on Fuzzy Logic in AI, pp. 114-125, 1993.

23. S. G. Goodridge, M. G. Kay, and R. C. Luo: "Multi-layered fuzzy behavior fusion for reactive control of an autonomous mobile robot", Proc. IEEE Intern. Conf. on Fuzzy Systems, pp. 579-584, Barcelona (Spain), 1997.

24. M. Skubic, S. Graves, and J. Mollenhauer: "Design of a two-level fuzzy controller for a reactive miniature mobile robot", Proc. IEEE Intern. Conf. on Industrial Fuzzy Control and Intelligent Systems, pp. 224-227, Houston, 1993.

25. P. Reignier: "Fuzzy logic techniques for mobile robot obstacle avoidance", Journal of Robotics and Autonomous Systems, pp. 143-153, 1994.

26. H. R. Beom and H. S. Cho: "A sensor-based obstacle avoidance controller for a mobile robot using fuzzy logic and neural network", Proc. Conf. on Intelligent Robots and Systems (IROS), pp. 1470-1475, 1992.

27. E. Tunstel: "Adaptive Hierarchy of Distributed Fuzzy Control: Application to behavior control of rovers", Ph. D. Thesis, Department of Electrical and Computer Engineering, University of New Mexico, 1996.

28. E. Tunstel: "Coordination of distributed fuzzy behaviors in mobile robot control", Proc. IEEE Intern. Conf. on Systems, Man, and Cybernetics, pp. 4009-4014, Vancouver (Canada), 1995.

29. E. Tunstel, H. Danny, T. Lippincott, and M. Jamshidi: "Adaptive fuzzy behavior hierarchy for autonomous navigation", Proc. IEEE Intern. Conf. on Robotics and Automation, pp. 829-834, Albuquerque, 1997.

30. E. Tunstel, T. Lippincott, and M. Jamshidi: "Behavior hierarchy for autonomous mobile robots: Fuzzy behavior modulation and evolution", Intern. Journ. of Intelligent Automation and Soft Computing, vol. 3, no. 1, pp. 37-49, 1997.

31. H. Seraji: "Traversability Index: A new concept for planetary rovers", Submitted to 1999 IEEE Intern. Conf. on Robotics and Automation.

32. D. B. Gennery: "Traversability analysis and path planning for a planetary rover", Submitted to Journal of Autonomous Robots, 1998.

33. R. Volpe and R. Ivlev: "A survey and experimental evaluation of proximity sensors for space robotics", Proc. IEEE Intern. Conf. on Robotics and Automation, vol. 4, pp. 3466-3473, San Diego, 1994.

34. K. M. Passino and S. Yurkovich: "*Fuzzy Control*", Addison-Wesley Publishing Company, 1997.

35. B. Bon and H. Seraji: "On-line collision avoidance for the Ranger telerobotic flight experiment", Proc. IEEE Intern. Conf. on Robotics and Automation, vol. 3, pp. 2041-2048, Minneapolis, 1996.

36. H. W. Stone: "Mars Pathfinder microrover: A low-cost, low-power spacecraft", Proc. AIAA Forum on Advanced Developments in Space Robotics, Madison, 1996.

37. E. Gat, R. Desai, R. Ivlev, J. Loch, and D. P. Miller: "Behavior control for robotic exploration of planetary surfaces", IEEE Trans. on Robotics and Automation, vol. 10, no. 4, pp. 490-503, 1994.

38. J. Morrison and T. Nguyen: "On-board software for the Mars Pathfinder microrover", Proc. Second IAA Intern. Conf. on Low-Cost Planetary Missions, Applied Physics Laboratory, Laurel, 1996.

39. L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe, and T. Litwin: "Mars microrover navigation: Performance evaluation and enhancement", Proc. IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems (IROS), pp. 433-440, Pittsburgh, 1995.

40. R. Volpe, J. Balaram, T. Ohm, and R. Ivlev: "Rocky 7: A next generation Mars rover prototype", Advanced Robotics, vol. 11, no. 4, pp. 341-358, 1997.

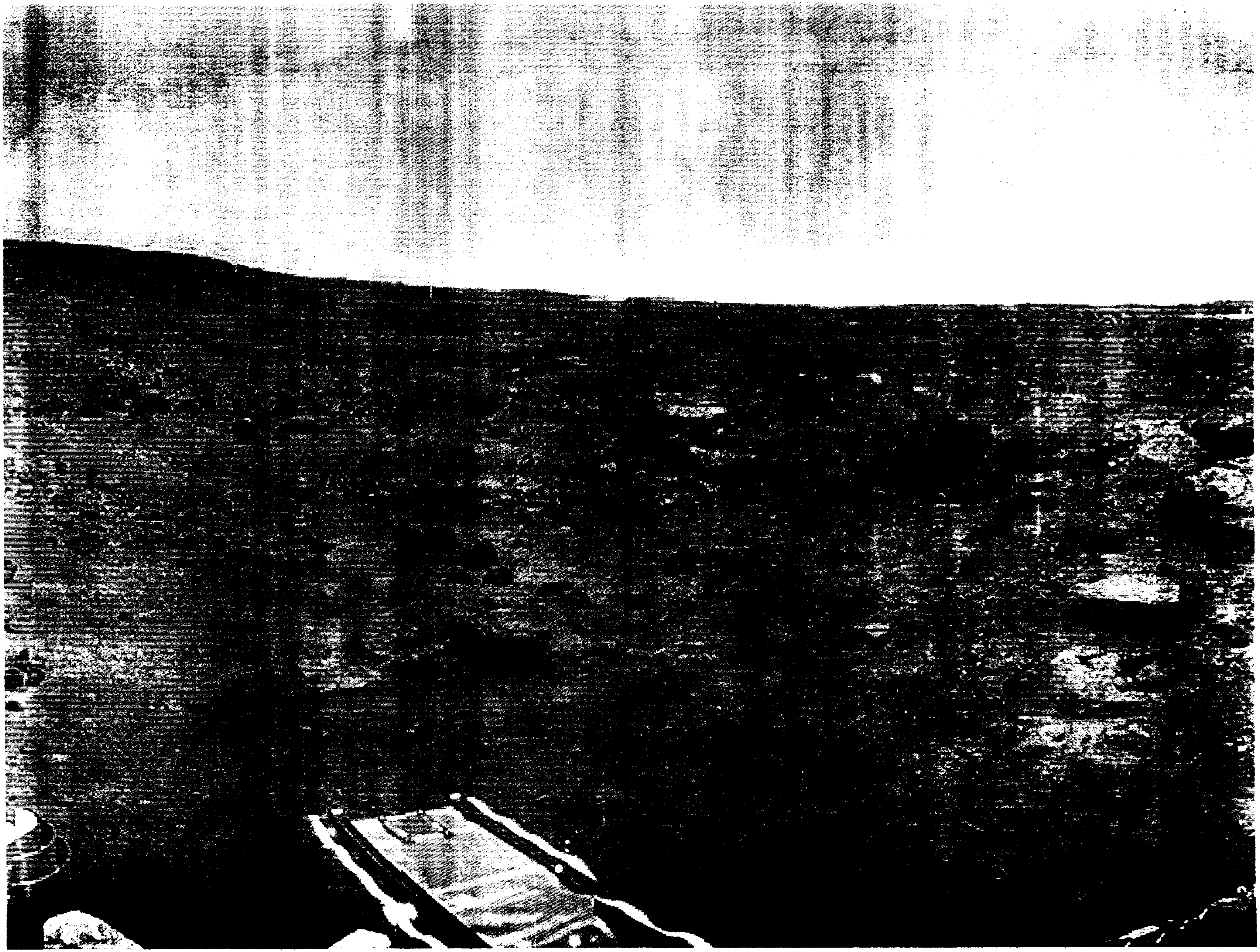


Figure 1. Photograph of the Martian terrain

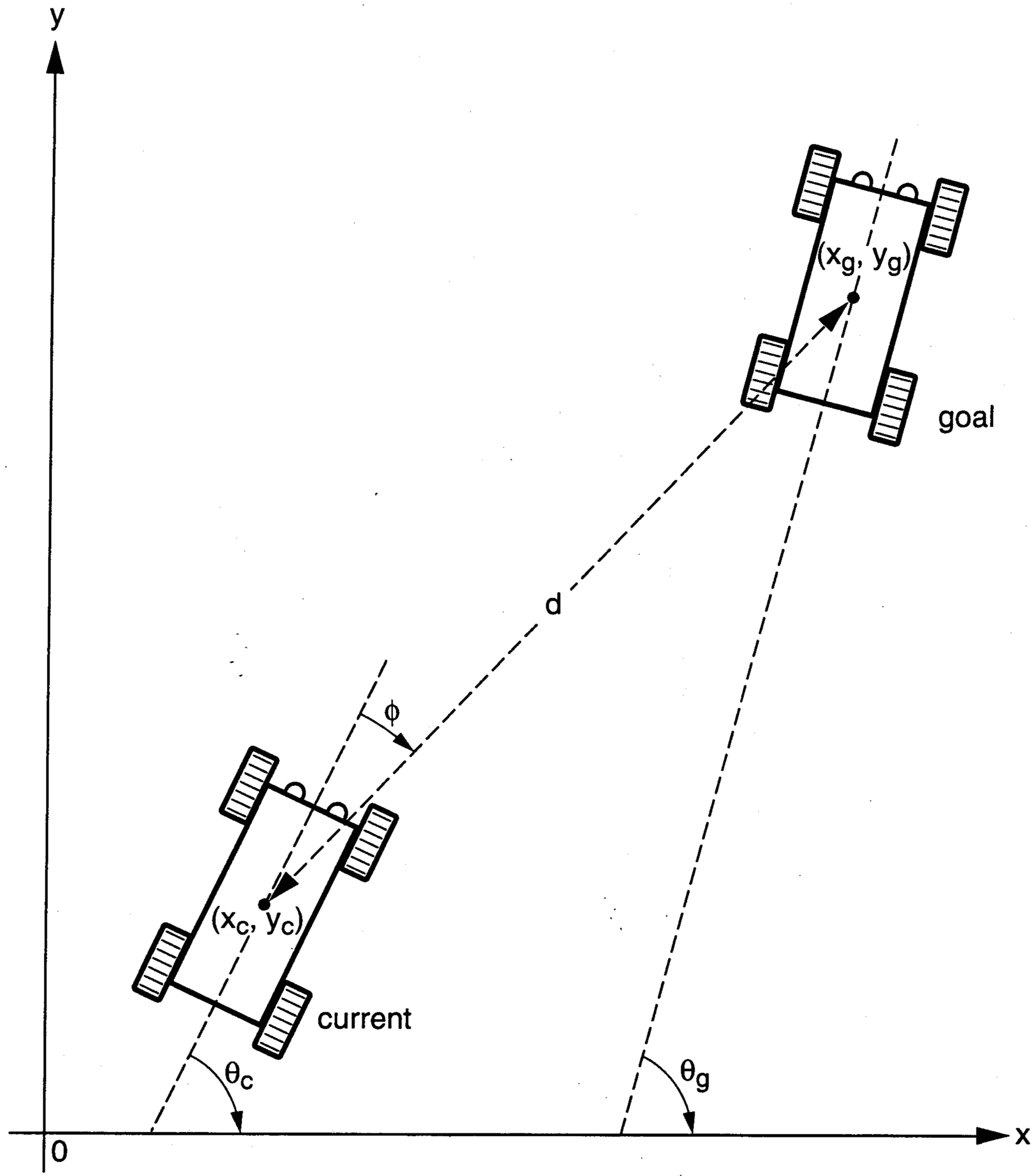


Figure 2. Definition of rover variables

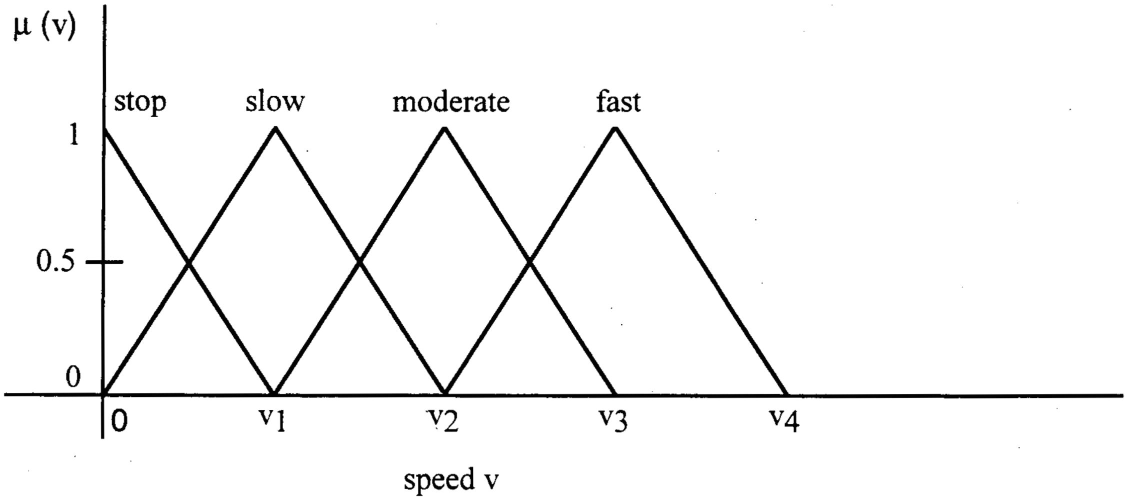


Figure 3a. Membership functions for speed v

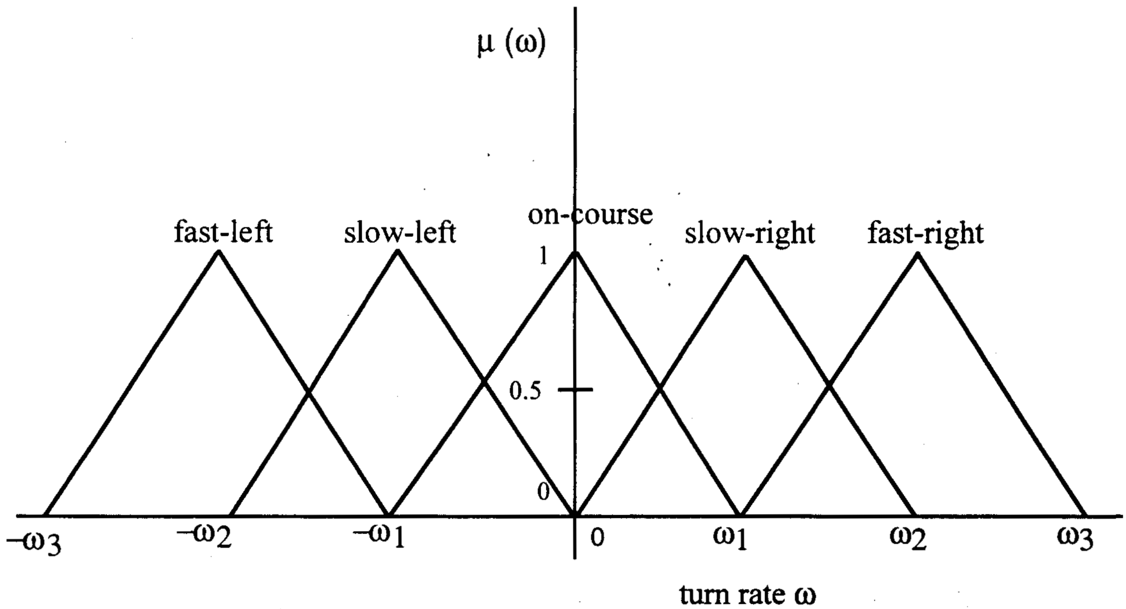


Figure 3b. Membership functions for turn rate ω

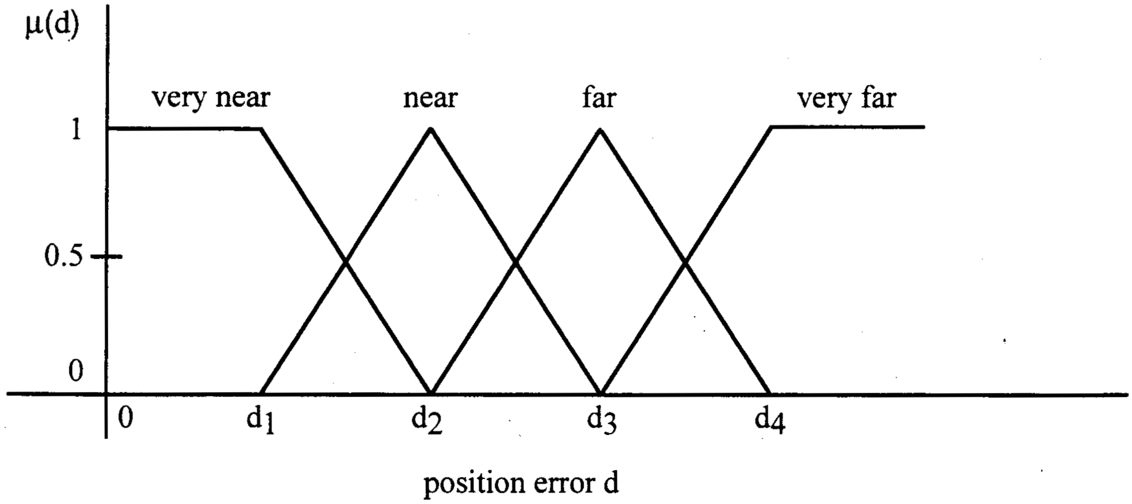


Figure 3c. Membership functions for position error d

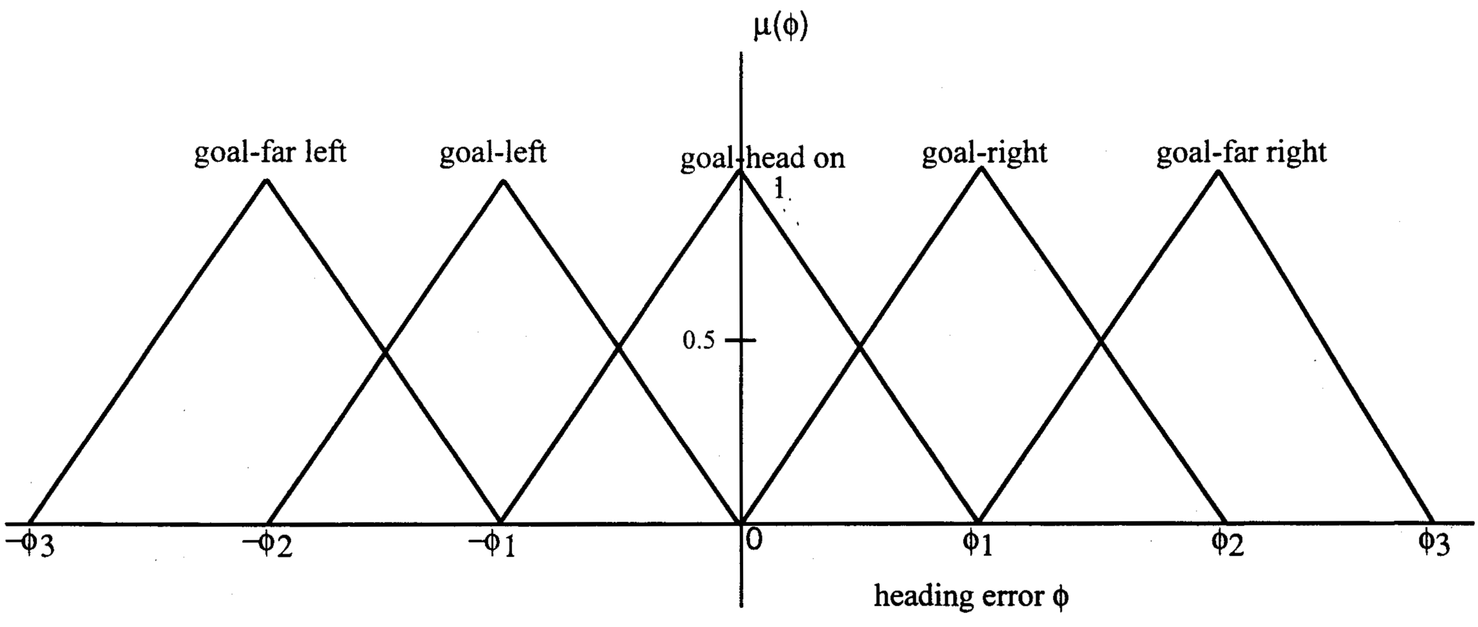


Figure 3d. Membership functions for heading error ϕ

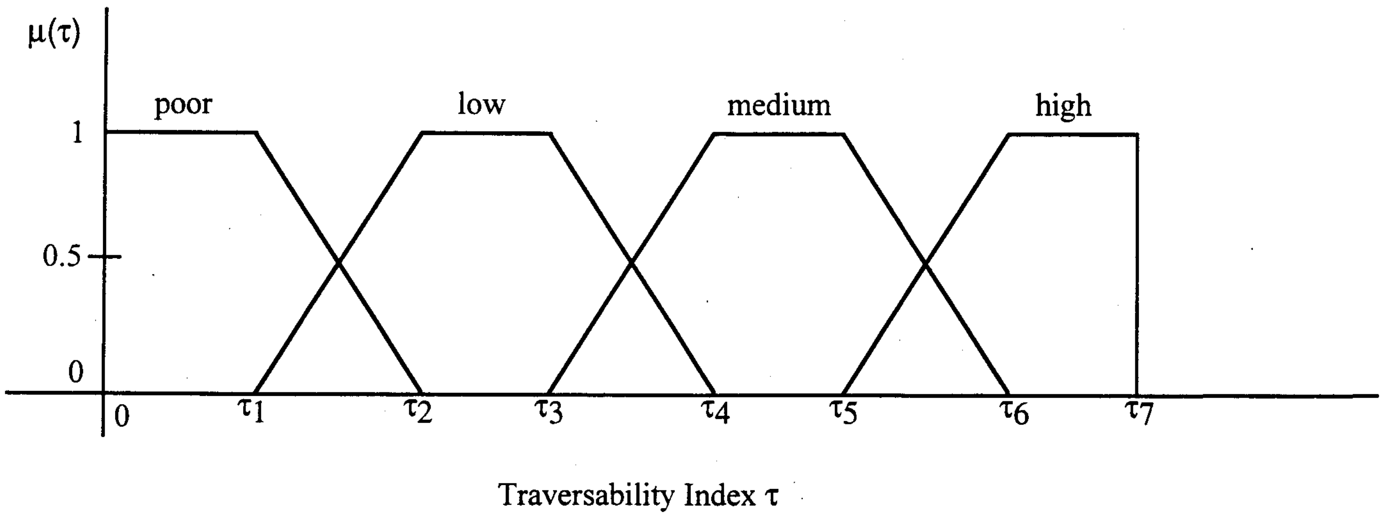


Figure 4a. Membership functions for Traversability Index τ

terrain slope α \ terrain roughness β	smooth	rough	bumpy	rocky
low	high	medium	low	poor
medium	medium	medium	low	poor
high	low	low	poor	poor
very high	poor	poor	poor	poor

Table 1. Rule set for Traversability Index τ

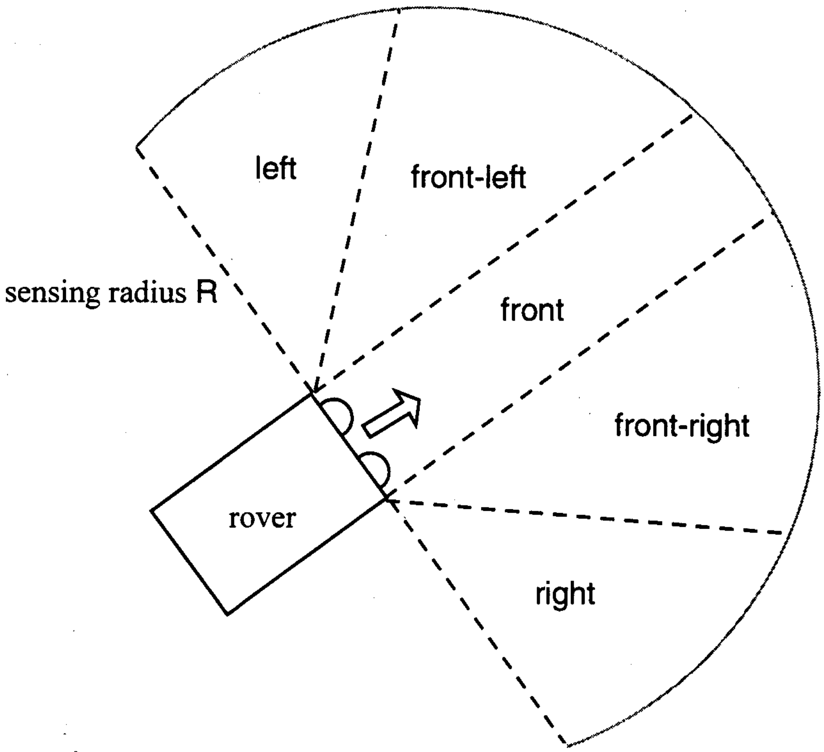


Figure 4b. Definition of traversable regions

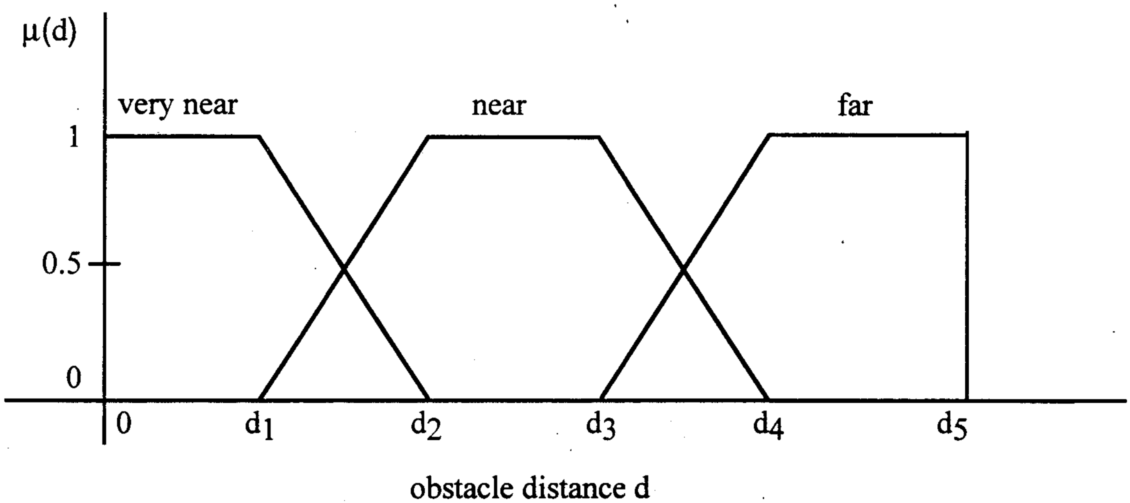


Figure 5. Membership functions for obstacle distance d

right obstacle distance d_r \ left obstacle distance d_l	very near	near	far
very near	slow-left	slow-left	slow-left
near	slow-right	slow-left	slow-left
far	slow-right	slow-right	slow-left

Table 2: Rule set for collision avoidance when d_f is NOT FAR

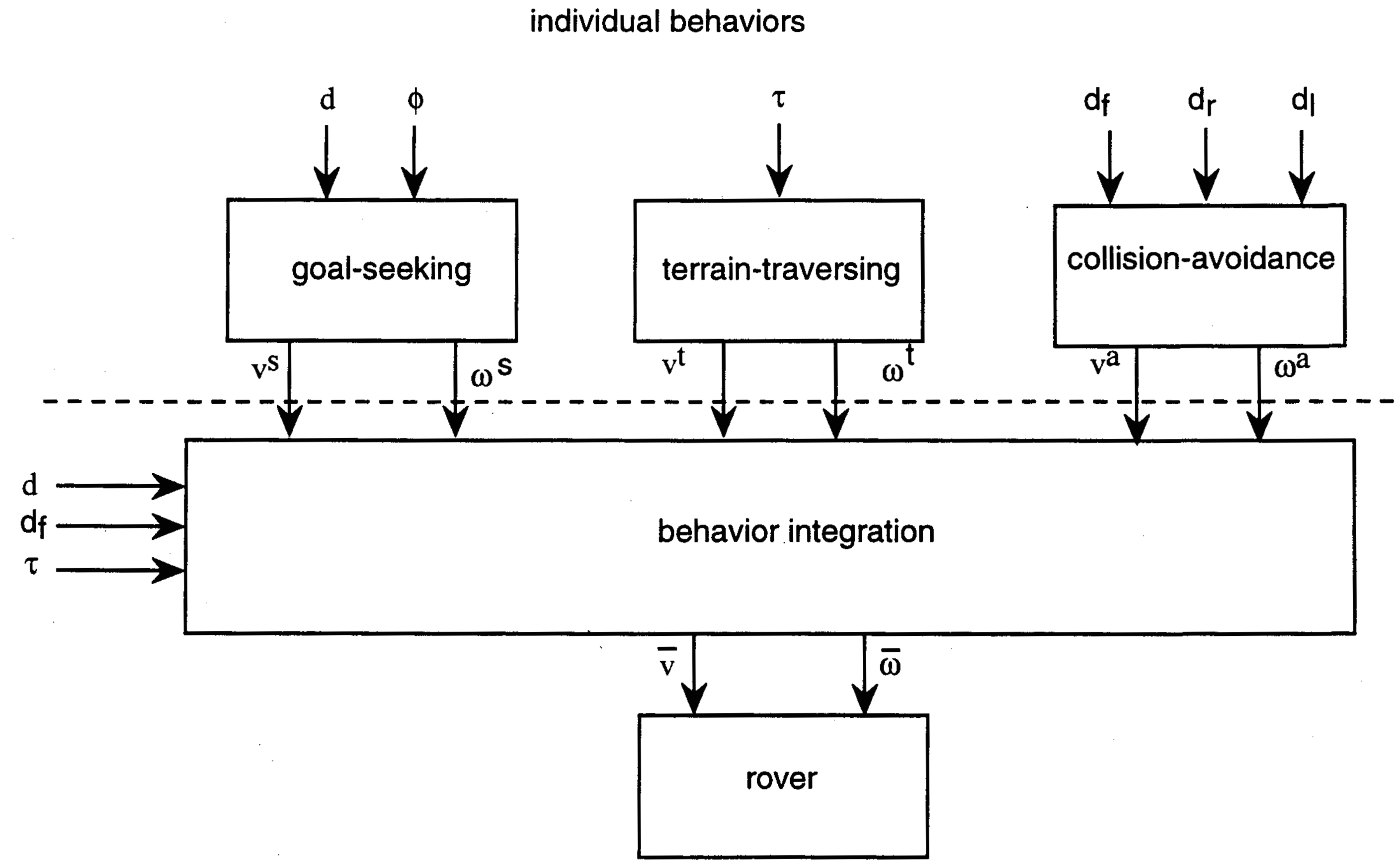


Figure 6a. Two-stage rover navigation strategy

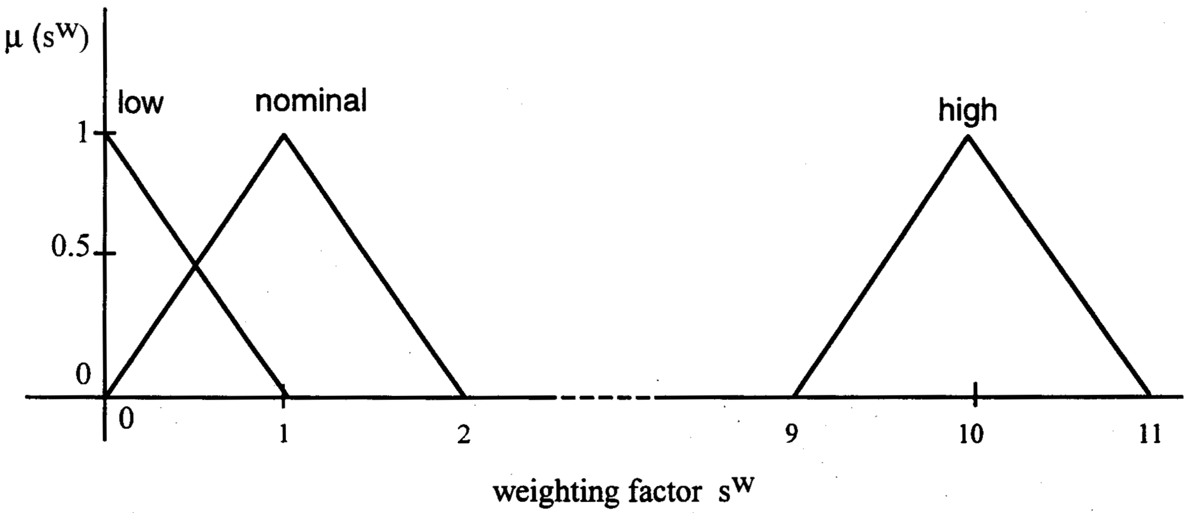
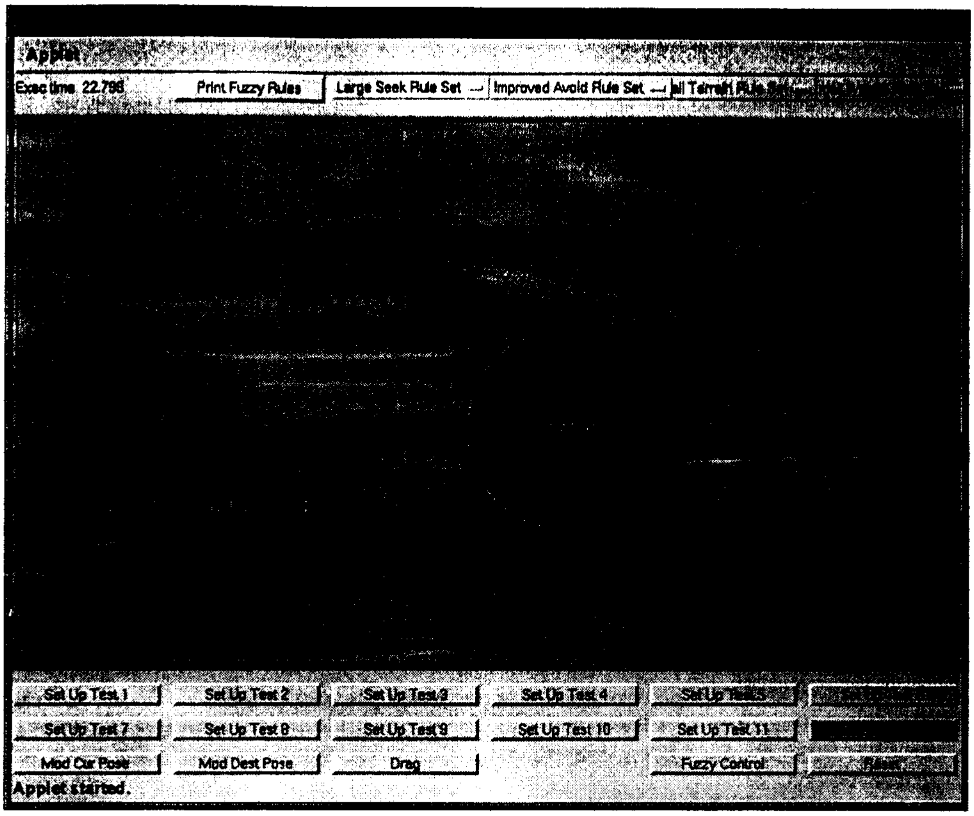
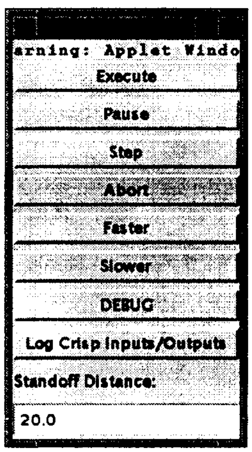
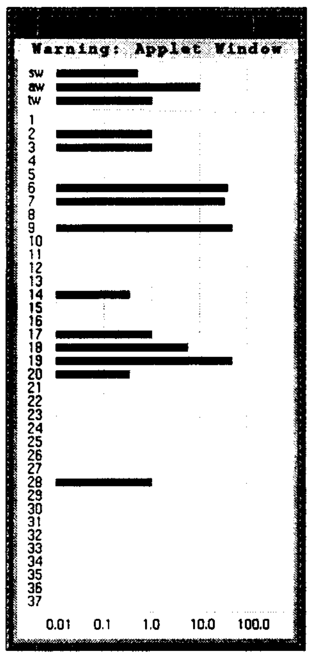


Figure 6b. Membership functions for weighting factors



(a) Principal window

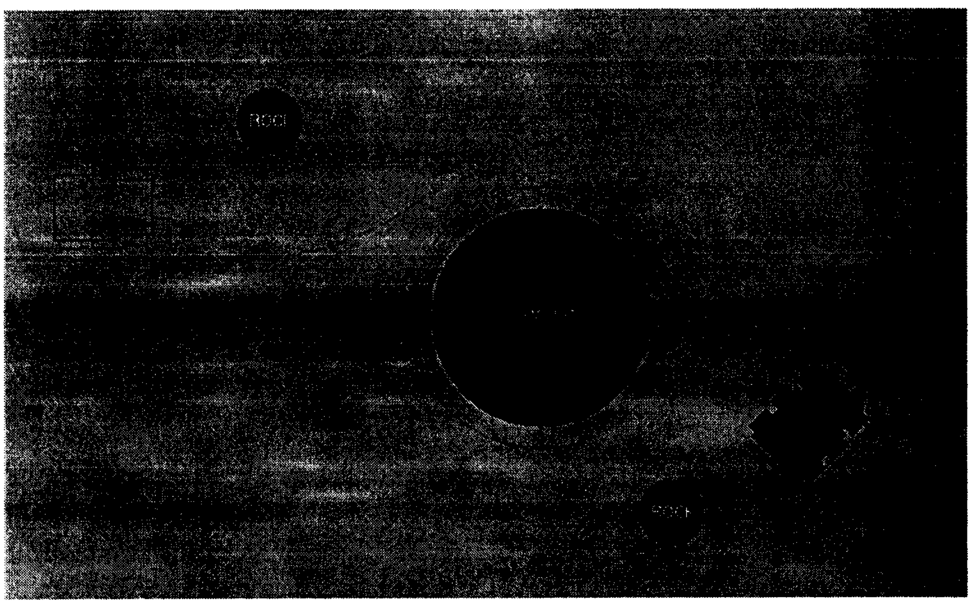




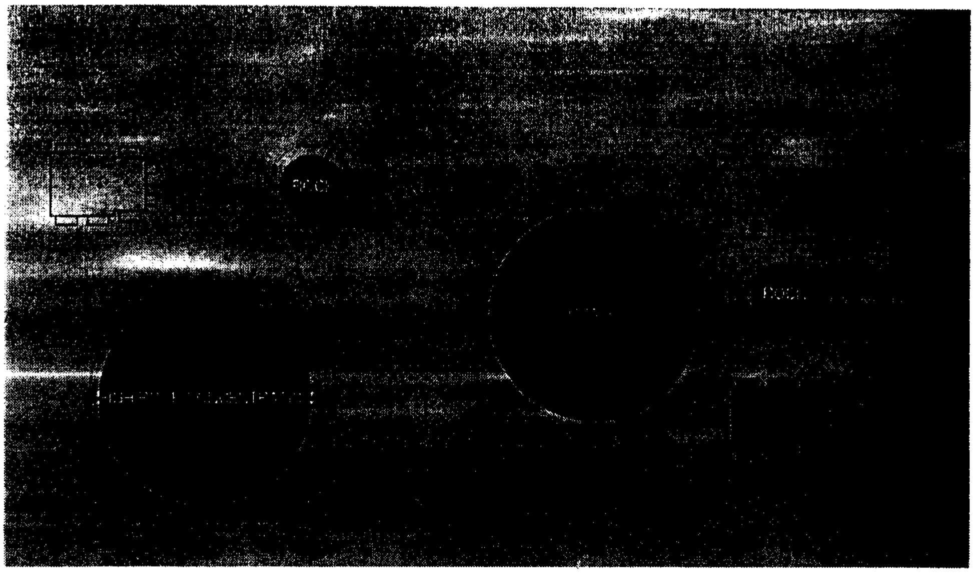
(b) Execution control and rule strength windows

Figure 7: RGS windows

(a) Case study one

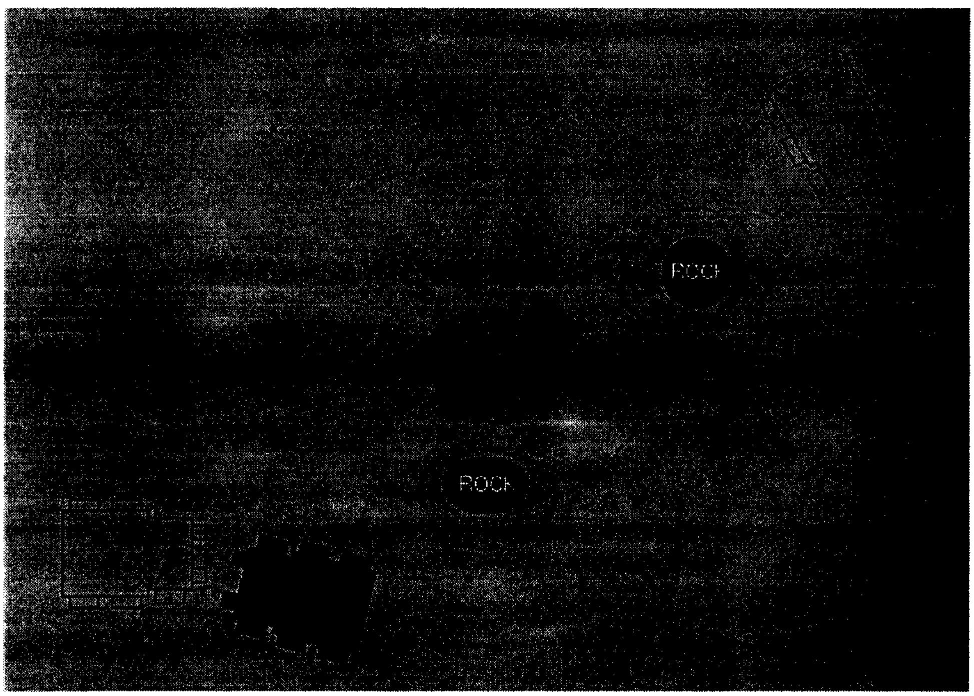


(b) Case study two

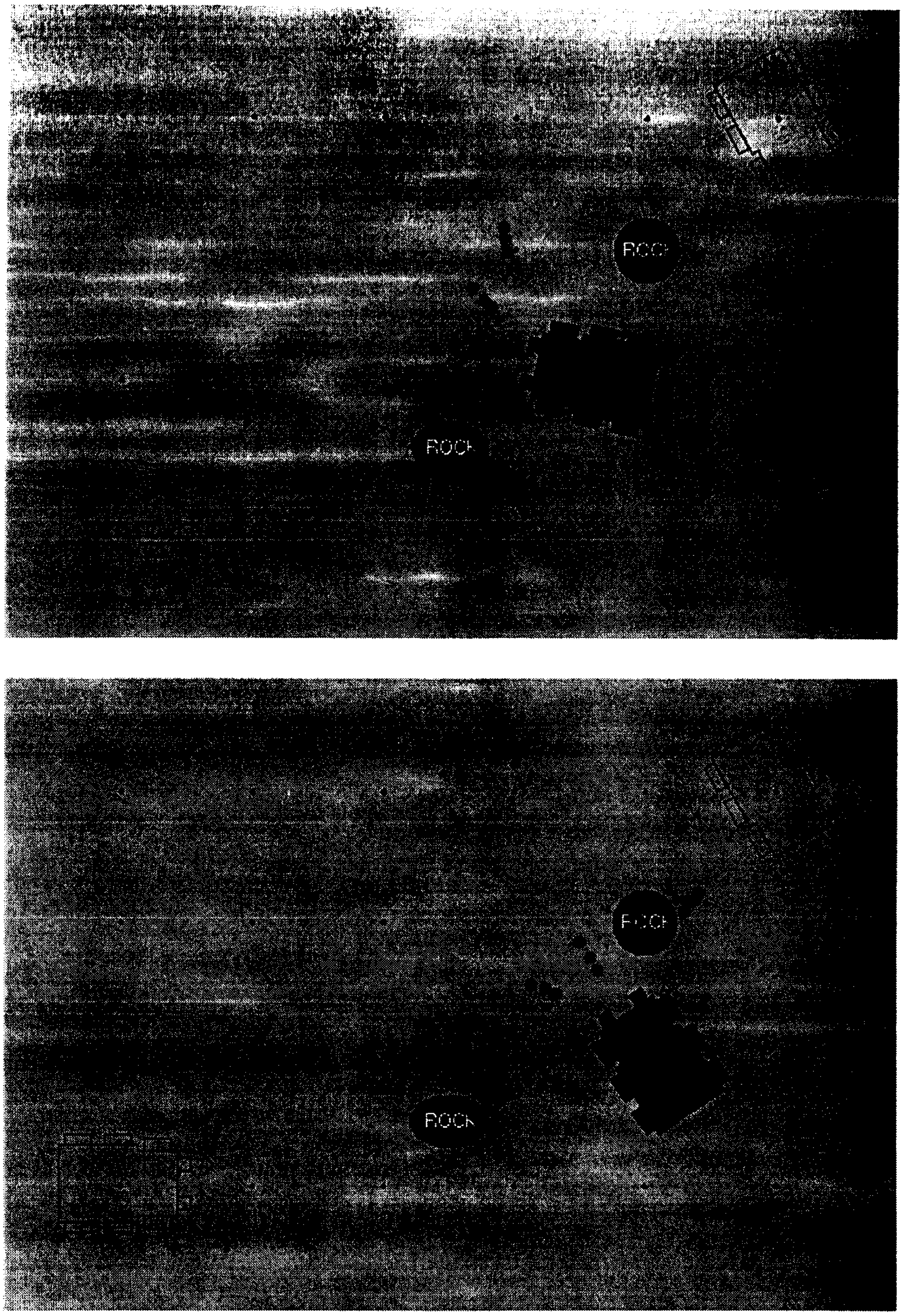


(c) Case study three

Figure 8: Rover paths using the fuzzy navigation strategy



(a) Fuzzy navigation algorithm



(b) Sojourner navigation algorithm

Figure 9: Rover paths for collision avoidance

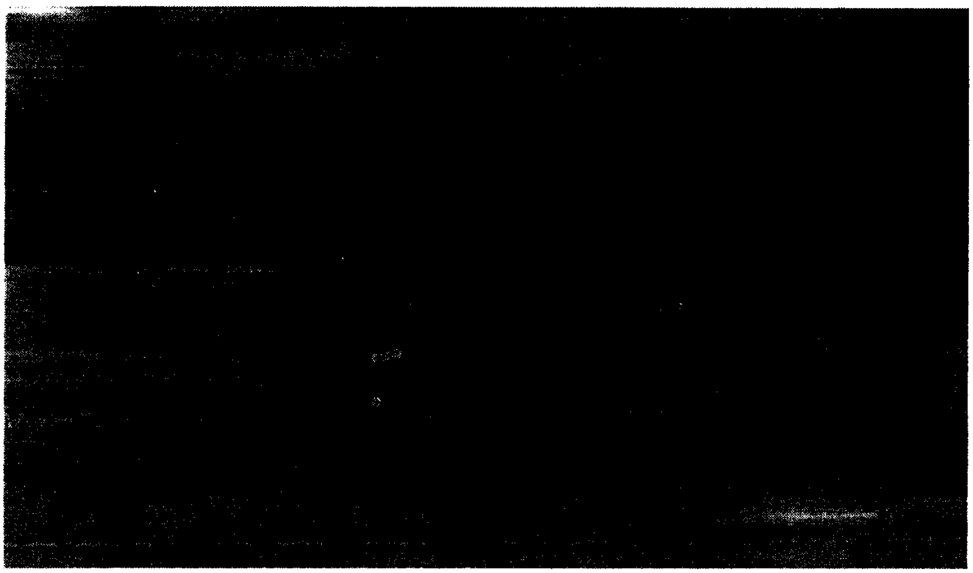


Figure 10: Rover path using the Sojourner navigation algorithm encountering a poorly-traversable terrain